Previous Doc Next Doc Go to Doc#
First Hit Fwd Refs

Generate Collection

L10: Entry 2 of 3 File: USPT Aug 12, 2003

DOCUMENT-IDENTIFIER: US 6606630 B1
TITLE: Data structure and method for tracking network topology in a fiber channel port driver

Application Filing Date (1):
20000821

Brief Summary Text (10):
Fibre Channel storage-area-network (SAN) nodes and switches, especially network nodes having multiple ports, must keep track of a variety of information about the network and resources available over the network. This information is used by each node to format and properly route frames onto and over the network.

Brief Summary Text (13):
When a command block references storage accessible over a Fibre Channel network, the driver must encapsulate the device level commands into one or more command frames, and for write operations one or more data frames. The driver may use the network information to determine header information and routing for the one or more fibre channel network frames, or packets, that implement the command.

Detailed Description Text (52):
Dynamic Control of Command Queue Depth

Previous Doc Next Doc Go to Doc#

Previous Doc Next Doc Go to Doc#
First Hit

☐ Generate Collection

L7: Entry 1 of 14 File: PGPB May 23, 2002

DOCUMENT-IDENTIFIER: US 20020061018 A1
TITLE: Switch transferring data using data encapsulation and decapsulation

Application Filing Date:
20010315

Detail Description Paragraph:
[0032] Memory interfaces 110, 112 and 114 can be located on switch 100 and allow for the separation of data and control information. Packet buffer memory interface (PBM) 110 handles packet data storage while the transmit queue memory interface (TXM) 112 keeps a list of packets to be transmitted and address table/control memory interface (ATM) 114 handles the address table and header information. Each of these interfaces, in one example of the invention, uses SSRAM that can be configured in various total amounts and chip sizes.

Detail Description Paragraph:
[0035] ATM 114 can be located on switch 100 and can have an external address table/control memory (not depicted) used to store the address table and header information corresponding to each 256 byte section of PBM 110. Address table/control memory allows up to 16K unique unicast addresses. The remaining available memory is used for control information. ATM 114, in this example, runs up to 133 MHz.

Detail Description Paragraph:
[0036] Switch 100, in one example of the invention, has a Flow Control Manager 116 that manages the flow of packet data. As each port sends more and more data to the switch, Flow Control Manager 116 can monitor the amount of memory being used by each port 102(1)-102(12) of switch 100 and the switch as a whole. In this example, if one of the ports 102(1)-102(12) or the switch as a whole is using up to much memory, Flow Control Manager 116 will issue commands over the ATM Bus requesting the port or switch to slow down and may eventually drop packets if necessary.

Detail Description Paragraph:
[0040] Address Manager (AM) 122 can manage the address table including learning source addresses, assigning headers to packets and keeping track of these addresses. In one example of the invention, AM 122 uses aging to drop addresses that have not been used for some specified time period or sequence of events.

Detail Description Paragraph:
[0055] If the packet is "good", the port writes the header information to the ATM memory through the ATM Bus and ATM 114. The port also sends a RECPE_COMPL command over the ATM Bus signifying that packet reception is complete. Other information is also sent along with the RECEP_COMPL command such as the start address and filtering table which indicates which ports the packet is to be sent out on. For example, a filtering table having a string such as "011111111111" would send the packet to all ports except port 1 and would have a count of 11. The count simply is the number of ports the packet is to be sent.

Detail Description Paragraph:

[0063] If the port is busy, the RECEP_COMPL command information is transferred to TXM Memory through the TXM Bus and TXM 112. The TXM memory is simply a queue of packets to be transmitted. TXM Memory is allocated on a per port basis so that if there are ten ports there are ten queues within the TXM Memory allocated to each port. As each of the ports transmitters becomes idle, each port will read the next RECEP_COMPL command information stored in the TXM Memory. The TX FIFO of port 102 (12) will receive, as part of the RECEP_COMPL command information, a start pointer which will point to a header in ATM memory across the ATM Bus which in turn points to the location of a packet in the PBM Memory over the PBM Bus. The port will at this point request to load the packet into the transmit (TX) FIFO of port 102(12) and send it out through the MAC and PHY of port 102(12).

Detail Description Paragraph:
[0081] In one example of the present invention as depicted in FIG. 6, data encapsulation/decapsulation is performed on the IEEE 802.3 Ethernet packet without changing the packet length thereby maintaining data stream performance. In this example, the Frame Check Sequence (FCS) of the 802.3 packet will be encapsulated with the Source ID on the transmit side and the FCS of the 802.3 packet will be decapsulated on the receive side to identify the encapsulated Source ID.

Detail Description Paragraph:
[0083] In step 610, information is encapsulated into the FCS field of the data packet. In this example, the Switch ID is encapsulated into the FCS field in order to prevent continuous looping of the packet in a stacking environment.

Detail Description Paragraph:
[0084] In step 620, the data packet encapsulated, in this example, with the originating Switch ID in the FCS field of the data packet, is transmitted to a second switch, SW2.

Detail Description Paragraph:
[0087] FIG. 7 is a table made up of three columns. The first column is labeled Syndrome and represents all the possible values resulting from a CRC calculation on the entire data packet. The second column is labeled Data and represents information that is to be transferred with the data packet. The third column is labeled Distance Vector and represents the actual data encapsulated into the FCS filed of the data packet.

Detail Description Paragraph:
[0088] In the present example, if the originating switch were switch 6, the data transferred with the data packet would be 6. From the table depicted in FIG. 7, the distance vector that should be encapsulated in the FCS field of the data packet should be Ox 06_06_06_06. Therefore, if a CRC polynomial is applied to the entire data packet, a syndrome of Ox 7C_80_26_02 should be returned. Although this value is not the IEEE defined value of Ox C7_04_DD_7B, the value of Ox 7C_80_26_02 is correct since the FCS field of the data packet has been encapsulated with the Distance Vector Ox 06_06_06_06 indicating that the originating switch is switch 6.

Detail Description Paragraph:
[0090] FIG. 8A is an illustration of one example of an Encapsulating Apparatus 800. In this example, Encapsulating Apparatus 800 has an XOR module 802 and an Encoding Code Book 804. In one example of the invention, Information to be encapsulated into the FCS field of a data packet is inputted through Encoding Code Book 804. If the Information to be encoded was data such as the number 5, the distance vector would be Ox 05_05_05_05 (see FIG. 7 where a data value of 5 has a distance vector of Ox 05_05_05_05. In this example the original FCS and the distance vectors are used as input to XOR module 805. In this example if the Original FCS were Ox 60_60_60_60 the Transmit FCS would be Ox 65_65_65_65. Information is now encapsulated into the Transmit FCS.

Previous Doc Next Doc Go to Doc#
First Hit

Generate Collection

L7: Entry 2 of 14 File: PGPB Jul 26, 2001

DOCUMENT-IDENTIFIER: US 20010009547 A1
TITLE: Data communications system

Application Filing Date:
20010123

Summary of Invention Paragraph:
[0008] Each DIF sequence is composed of 150 DIF blocks and has a structure where a regular pattern consisting of 15 pieces of video data (v) continuing after one piece of audio data (a) following six frame header information (f), is repeated, as shown in FIG. 1B.

Summary of Invention Paragraph:
[0011] For example, if digital video data are transmitted via a local bus based on the IEEE1394 standard, an IEEE1394 packet is generated by attaching both a prescribed header (CIP (Common Isochronous Packet) header and synchronous (ISO) header) and CRC to six DIF blocks, as shown in FIG. 2A and digital video data are transmitted in transfer units of these IEEE1394 packets.

Summary of Invention Paragraph:
[0021] A frame thinning-out process is performed by this digital video transmitting unit 12. Simultaneously, as shown in FIG. 4, header information consisting of an IP header, UDP (User Datagram Protocol) and an application header is attached to an IEEE1394 packet to be transmitted, and a packet encapsulated in an IP packet is transmitted to the Internet via an Internet adapter 413.

Summary of Invention Paragraph:
[0024] In this data communications system, an application header, including both an adjustment parameter for indicating information about frame thinning-out, etc., and a sequence number is attached in the encapsulation process by the digital video transmitting unit 12, and an IEEE1394 packet stream is reproduced according to information in this application header in the decapsulation process by the digital video receiving unit 21.

Summary of Invention Paragraph:
[0025] The data communications system shown in FIG. 3 simply encapsulate digital video data in a UDP packet and transmits the data, and information inserted in the application header when data are encapsulated is limited to a few items, such as a sequence number, etc.

Summary of Invention Paragraph:
[0040] This IEEE1394 standard is a high-speed serial bus standard that is stipulated around both a physical layer and a data link layer, and stipulates a function to consecutively transfer a prescribed transfer unit in a specific cycle (synchronous transfer mode) and a function to transfer a control command from time to time to control equipment, such as a digital video camera, etc. (asynchronous transfer mode).

Summary of Invention Paragraph:

[0043] If particularly, animation data with audio data are transferred in real time in a synchronous transfer mode, the transmitting side node attaches a CIP (Common Isochronous Packet) header to transfer data in addition to a synchronous header, including the channel ID described above, generates an aisochronous packet shown in FIG. 6B and transmits the assigned aisochronous channel. A receiving side node identifies a packet to be received based on the channel ID included in the aisochronous packet and receives the transfer data.

Summary of Invention Paragraph:
[0044] The CIP header is provided with 16 bits of a time stamp field, and when transferring digital video data, the transmitting side node writes a time stamp for indicating a transmission time in one of a series of packets composing one frame of video data. The receiving side node adjusts timing based on this time stamp.

Summary of Invention Paragraph:
[0045] In this way, in the synchronous transfer mode of the IEEE1394 standard, an individual communications is identified by the channel ID included in the synchronous header, and data can be transferred while being synchronized by adjusting timing based on the time stamp in the CIP header regardless of the number of receiving side nodes.

Summary of Invention Paragraph:
[0046] However, in asynchronous transfer mode, after obtaining a right to use a bus, a transmitting side node generates an asynchronous packet (see FIG. 6C) by attaching an asynchronous header, including respective node IDs for indicating a transmitting node and a receiving node to transfer data, and transmits the packet to a bus.

Summary of Invention Paragraph:
[0047] Then, a receiving side node receives the packet addressed to him/her based on a receiving ID included in the header and transmits a reply packet (shown by symbol "ack" to a bus within a prescribed time period in FIG. 6A).

Summary of Invention Paragraph:
[0048] In this way, in an IEEE1394 asynchronous transfer mode, an individual communications is identified by the combination of a source ID and a destination ID included in the asynchronous header, and the arrival of a packet with transfer data is confirmed by a prescribed reply packet returned by the receiving side node.

Summary of Invention Paragraph:
[0055] If each of a transmitting node and a receiving node belongs to a different network, timing cannot be adjusted using a time stamp indicating a time in the network on the transmitting side without any modification on the receiving side even if the time stamp is written in the CIP header of a synchronous transfer mode and is transferred to the receiving node.

Summary of Invention Paragraph:
[0073] According to the first data communications system of the present invention, in a data communications system in which a transmitting side relay device and a receiving side relay device are connected to the first and second networks, respectively, for consecutively transferring prescribed transfer units at a specific transfer rate and with a specific transfer delay, and in which a data stream generated as a series of the transfer units is communicated as a series of datagram type packets, including the transfer unit, via the third network, the transmitting side relay device comprises additional information generation means for generating additional information, including information about relationship between each transfer unit and the data stream, based on the characteristics of the data stream, packet generation means for generating a packet by attaching the additional information when each transfer unit composing the data stream is inputted and by also attaching header information suitable for transfer in the

Previous Doc Next Doc Go to Doc#
First Hit Fwd Refs

Generate Collection

L17: Entry 2 of 7 File: USPT Sep 2, 2003

DOCUMENT-IDENTIFIER: US 6614796 B1
TITLE: Fibre channel arbitrated loop bufferless switch circuitry to increase bandwidth without significant increase in cost

Application Filing Date (1):
19981119

Detailed Description Text (132):
Priority escalates from none to low after a management programmable number of consecutive denials. If management software sets 0 as the number of denials, this escalation is disabled. After the trigger level is exceeded, the priority deny count is reset and a low-priority request bit is set for exactly one arbitration interval. This arbitration interval begins after the low priority trigger value has been exceeded and an IDLE primitive has been detected on the inbound port of the state machine. All subsequent remote requests will use low priority until another IDLE is detected, ending the arbitration interval.

Detailed Description Text (145):
Once the system is locked, external logic asserts a signal which sets the port multiplexers such that each port is partitioned from the system. During this time, each loop is broken, and IDLEs are driven onto the local loops in order to nullify ARBs that may still be present from hub mode arbitration. Next, the locking signal is deasserted, and, after a brief delay in which IDLEs are sent to the outbound port, the switching chips transition to the monitoring state and switch mode operation commences.

Detailed Description Text (162):
The ports on every switch actively participate on their loops when they are facilitating a remote tenancy. The two primitives RRDY and CLS are used according to FCAL net interframe gap policy. The specific policy that the switch chips use is that if a current primitive is a fill word and two consecutive fill words have previously been detected, then insertion of the RRDY or CLS may take place. For purposes of this policy, fill words are defined as ARB(x) and IDLE primitives.

Detailed Description Text (164):
Fill words are generated and inserted onto the local loop when the LPSM is not in the LPSM_MON state (hereafter the states of the LPSM will be referred to by the acronym that follows LPSM_in FIG. 11). Rules governing fill word generation ensure that the current arbitration state of each loop is maintained independently. The problem for each switch chip when it is coupled to a source loop, i.e., the loop having the source node, is that it must send fill words that are relatively innocuous. An NL_port in an OPEN state will transmit ARB(OF) which it uses to determine if any other ports on the loop are currently participating in arbitration. If the open NL_port receives ARB(F0) in return, the NL_port has the option of retaining the loop arid sets it current fill word to IDLE, thereby resetting the arbitration interval. The LPSM needs to send a management programmable low priority ARB that does not possess the potential to disturb the local loop access state like ARB(OF) does. For this purpose, the ARB(F7) is suitable, as it is only used by a non-participating port to quiesce the loop prior

to sending a loop initialization primitive LIP in order to receive a valid AL_PA.

Detailed Description Text (169):
The most common primitives that the switch chips must deal with are: IDLE, RRDY, ARB, OPN, SOF, EOF, CLS and LIP. The other primitives that may be encountered include: NOS, OLS, LR, LRR, MRK, LPB AND DHD. The handling of these other primitives is described below.

Detailed Description Text (170):
NOS, OLS, LR AND LRR PRIMITIVES

Detailed Description Text (171):
When NOS or OLS primitives are detected, on the inbound port of a loop, the OLD_PORT detect flag of the port is set so that the management software can detect the condition. LR and LRR primitives will be fed back onto the local loop, but not through the backplane.

Detailed Description Text (179):
The protocol bus is the medium by which the scoreboards for each switch chip are kep current. It also serves to communicate switch connection requests and responses between switch ports. The protocol bus is defined as an 18 bit bidirectional data bus named PBD, a request output PBREQ0 for each switch chip, a grant input named PGGRNT1 to each switch chip, a shared bus idle input signal PBIDLE1, and shared frame available input signals named PBFRM1. The protocol bus runs asynchronously with respect to the 106.25 MHz core circuitry to which it interfaces. It is only necessary to provide a clock of less than 50 MHz for the bus to function properly.

Detailed Description Text (184):
Chip remote request frames have a 0.times.0 pattern in the LCL field shown at 300 in FIG. 14A while chip local requests place a 0.times.3 in this field. This difference in format allows switch chips to update channel allocation scoreboards only on chip remote requests. If a chip remote request hits a channel that has been placed in an age list, a channel idle scoreboard for the channel owned by the destination port is cleared; otherwise, a need counter is incremented as will be explained in the next section.

Detailed Description Text (186):
A responder channel idle frame having the format of FIG. 14C is driven on the protocol bus when a backplane channel completes a transaction. In the default system configuration, the channel is retained by the responder until the channel is required by another responder, thus reducing setup overhead if the destination node that just finished a tenancy over the channel is a frequently accessed destination. When an idle frame is received, each switch chip updates its: channel idle scoreboard to indicate the channel is available.

Detailed Description Text (190):
The example given herein for, the configuration of the backplane has 14 separate backplane data channels, three of which may be dedicated to other uses such as broadcast. Channels are allocated by destination ports and remain assigned to them for as long as possible. A channel scoreboard indicates if any channels are free and is used to hold off any pending responses from ports which are not already connected to a data channel. If this is not the case, a need counter is incremented. When this need counter exceeds the physical number of data channels, then the backplane attempts to free up a channel while the destination port attempts to obtain a connection grant response from its LPSM. All data channel connections (identified by response frames on the protocol bus) are stored in the age list which indicates the channels that have been held the longest and the port that currently own those channels. When the need counter exceeds the number of data channels, each switch chip consults its age list and selectes the channel to be relinquished. Idle channels are broadcast on the protocol bus when the camp list of

the port owning the channel empties completely. The oldest owned channel is freed by the switch chips that owns the channel and all switch chips update their scoreboards to reflect the new state. For every channel that is freed, the need count is decremented by one.

Detailed Description Text (191):
Channels can also be freed in blocks of size greater than one. The size of the block of channels freed is determined by management-programmable parameter. The switch chip management logic can also instruct the backplane to always free a channel when it becomes idle, rather than only freeing the channel when the need arises.

CLAIMS:

3. The apparatus of claim 1 wherein said crossbar switch means comprises: a crossbar switch; a routing table coupled to said protocol bus and containing data indicating on which Fibre Channel Arbitrated Loop each node coupled to said switch is located or which port means must be used to send data to or receive data from each said node; a scoreboard means coupled to said protocol bus for storing data regarding which of said port means coupled to said destination nodes are busy and which are idle and for updating said data based upon the data content of messages on said protocol a fairness token bus coupled to each port means; and a bidirectional bus coupling said crossbar switch to each said port means for carrying data frames and primitives of said loop tenancies between said crossbar switch and said port means; and wherein each said port means comprises: a learning half bridge having an input port and an output port for coupling to the transmit and receive channels of a Fibre Channel Arbitrated Loop or a single Node Loop port which is part of a node and having a port coupled to said protocol bus, and having circuitry to transmit on said protocol bus messages regarding the busy or idle status of the Fibre Channel Arbitrated Loop or a single Node Loop port to which said learning half bridge is connected, and having a port coupled to said fairness token bus and circuitry coupled to said port for sending and receiving a fairness token, and said learning half bridge having circuitry coupled to said bidirectional bus for sending and receiving primitives and data frames of said loop tenancies to and from said crossbar switch.

7. A Fibre Channel Arbitrated Loop Switch, comprising: a fairness token bus; a protocol bus; a backplane data path capable of supporting multiple bidirectional data paths; a plurality of switch chip circuits, each comprising: a memory containing routing table data; a memory containing scoreboard table data; a learning half bridge front end circuitry which includes an FCAL port comprised of an input port and an output port for coupling to the transmit and receive lines of a Fibre Channel Arbitrated Loop or a single Node Loop port which is part of a computer or a computer peripheral device, said computer or computer peripheral device referred to herein as a node, each learning half bridge including circuitry to implement the Fibre Channel Arbitrated Loop protocol to communicate with nodes coupled to said FCAL port and to receive connection requests from said protocol bus, and including circuitry coupled to said FCAL port to arbitrate for control of said Fibre Channel Arbitrated Loop coupled to said FCAL port, if necessary, using said Fibre Channel Arbitrated Loop Protocol when a connection request is received at said switch chip circuit from another switch chip circuit, and to send message data on said protocol bus indicating the status of said FCAL port as busy or idle and to learn from message traffic on said protocol bus which other switch chip circuits are coupled to other nodes coupled to said Fibre Channel Arbitrated Loop Switch and store that data in said memory containing said routing table data and to learn from said message traffic on said protocol bus which said FCAL ports of other switch chip circuits are busy and which are idle and store that data in said memory storing scoreboard table data; streaming back end crossbar switch circuitry coupled to said backplane implementing a slice of a distributed crossbar switch and functioning to connection requests from said learning half bridge circuits to use

Previous Doc Next Doc Go to Doc#
First Hit Fwd Refs

Generate Collection

L17: Entry 4 of 7 File: USPT Jan 11, 2000

DOCUMENT-IDENTIFIER: US 6014370 A
TITLE: Apparatus for bridging between fibre channel networks and ATM network

Application Filing Date (1):
19990218

Brief Summary Text (10):
FC device 501 in the FC network, detecting a normality recovery in the FC link 502, performs initialization of FC link 502. Initialization process and abnormal detection/notification in the FC network are carried out by the receive/send actions of the ordered sets of codes generated in the encoding/decoding layer in the FC link. In more detail, the FC device 501 sends an special code OLS to the FC device 501' to indicate that it is in an "offline" state ready to carry out the initialization process. Upon receiving OLS code, FC device 501' sends a link reset code LR to FC device 501 to indicate that initialization is being performed. Upon receiving a link reset code LR, FC device 501 sends a link reset response code LRR to the FC device 501', as the response code to link reset code LR. When the link reset response code LRR is received at the FC device 501', the FC device 501' start sending a code Idle, indicating that it is now in a state to enable to transfer data. Upon receiving the code Idle, the FC device 501 similarly becomes ready to transfer data, and start sending a code Idle. Initialization of the FC link 502 is thus completed.

Detailed Description Text (7):
In the FC link init-protocol, the apparatus 100 first sends out an offline code (OLS), and waits for the arrival of an FC link reset code (LR) (S204, 205). When the link reset code (LR) is received from the FC link, it starts sending an init-start cell to the ATM network every periodic interval (T) (S206).

Detailed Description Text (9):
When an init-complete cell is received from the ATM network, the FC link init-protocol is restarted by taking the first step to stop sending the init-start cells to the ATM network, which has been carried out at periodic intervals (T) (S210). Next, after sending an FC link reset code (LR), the process in a standby state until an FC link reset response code (LRR) is received from the FC link (S211, S212). If the FC reset response code (LRR) is received from the FC link, a special code (Idle) is sent to the FC link, and the process becomes idle until the status code (Idle) is received from the FC link. When the status code (Idle) is received from the FC link, it indicates that the FC link init-protocol has been completed.

Detailed Description Text (14):
When the FC switch 302 receives the offline code (OLS), it detects that an abnormal event has taken place in the network so that the inter-network bridging apparatus 301 is in the init-start state. In response to this detection, FC switch 302 sends a link reset code (LR) to indicate starting of FC link initialization.

Detailed Description Text (15):
The inter-network bridging apparatus 301 which receives the link reset code (LR), periodically sends an init-start cell to the ATM network 305 every interval (T),

until an init-complete cell is received from the ATM network 305. Also, the inter-network bridging apparatus 301 withholds sending a link reset response code (LRR) as the response to the link reset code (LR). It should be noted here that, in the conventional method, a link reset response code (LRR) would have been sent to FC switch 302 at this point. Therefore, FC link init-protocol for FC link 302 is temporarily suspended in the present method.

Detailed Description Text (16):
In the meantime, the inter-network bridging apparatus 301' receiving an abnormal notification cell from the ATM network 305 sends an offline code (OLS) to FC switch 302'. FC switch 302' receiving the offline code (OLS) sends a link reset code (LR) to the inter-network bridging apparatus 301', in response. Inter-network bridging apparatus 301' receiving the link reset code (LR) periodically sends an init-start cell to the ATM network 305 every interval (T) until an init-complete cell is received from the ATM network 305. As before, the inter-network bridging apparatus 301' withholds sending a link reset response code (LRR), which would have been sent at the point to FC switch 302' in the conventional method, until an init-complete cell is received from the ATM network 305. Accordingly, the init-protocol of FC link 302' is suspended temporary at this point. The init-start cell sent by the inter-network bridging apparatus 301' is not received by the inter-network bridging apparatus 301 until the abnormality in the ATM network 305 is recovered.

Detailed Description Text (21):
Each of the inter-network bridging apparatuses 301 and 301' which received the init-complete cell send a link reset code (LR) to the FC switches 302 and 302' to notify the FC link to start initialization respectively, thereby the suspended initialization process of FC links 303 and 303' are restarted. FC switches 302 and 302' which received the link reset code (LR) send respective responses in the form of link reset response code (LRR). Upon receiving the reset response code (LRR), the inter-network bridging apparatuses 301 and 301' are now in a state ready to transfer data, and both apparatuses send a special code (Idle) to respective FC switches 302 and 302'. FC switches 302, 302' receiving the status code (Idle) are now in a state ready to transfer data, and each switch sends a special code (Idle) to the respective inter-network bridging apparatus 301 and 301'. At this point, initialization processes for both FC links 303, 303' are completed.

Previous Doc Next Doc Go to Doc#

Previous Doc Next Doc Go to Doc#
First Hit Fwd Refs

Generate Collection

L17: Entry 6 of 7 File: USPT Jun 10, 1997

DOCUMENT-IDENTIFIER: US 5638518 A
TITLE: Node loop core for implementing transmission protocol in fibre channel

Application Filing Date (1):
19941024

Brief Summary Text (38):
Two possible FC frame formats with FC frames being separated from each other by at least six four-byte IDLEs.

Detailed Description Text (11):
FIG. 6 illustrates the architecture of the Arbitrated Loop 14 of FIG. 3. A Loop State Machine 60 performs loop initialization protocol, loop arbitration, and detects all ordered sets pertinent to the Arbitrated Loop. The Loop State Machine is used to acquire Arbitrated Loop physical addresses at Power On, and after loop initialization the NL.sub.-- Port goes into a monitoring state. When the loop is idle and the NL.sub.-- Port wants to communicate with another NL.sub.-- Port, loop arbitration protocol is performed by the Loop State Machine 60. When arbitration is won by any loop port, all the intermediate loop ports act as repeaters of transmission words. In this case, a loop buffer is used to provide the elasticity to re-time the data and send data to the transmitter. The Loop State Machine also looks for the primitive sequence which indicates that the present Arbitrated Loop connection is over.

Detailed Description Text (29):
FCS Idles and Primitive Sequences

Detailed Description Paragraph Table (2):
_____ __ Decode Word Signals The Decode Word Module Interface receives all incoming signals from the Fiber Optic Module (FOM). This includes recovered receive data, data width control, and status from both the transmit and receive FOMs. FDB.sub.-- IN[19:0] FOM Data In Input These signals contain two 10-bit FCS characters from the FOM. See subsection entitled "Decode.sub.-- word Interface Formats" on page 4-24 for the pin format. BYTE.sub.-- SYNC FOM Byte Sync Input Asserting this signal indicates that there is a Special Character (the first character of all ordered sets) or a comma to be read on FDB.sub.-- IN[19:0]. The FOM should assert this signal HIGH every time it detects a comma character. LASER.sub.-- STATUS FOM Transmitter Laser Fault Input Asserting this signal HIGH indicates that the FOM transmitter is detecting no laser output. The Receive passes this signal along as the STATUS.sub.-- REG10 output. LINK.sub.-- STATUS FOM Receiver Loss of Light (Open Input) Asserting this signal HIGH indicates that the FOM receiver is detecting no signal. This signal is an input into the Receive State Machine. The Receive passes this signal along as the STATUS.sub.-- REG9 output. UL.sub.-- SELECT 10 bit upper/lower select input When using 10bit FOMs, this input selects whether the 10bit FOM input to the Receive is received in the upper 10bits or the lower 10bits of the 20bit FDB.sub.-- IN[19:0] interface. 10.sub.-- 20.sub.-- SELECT 10bit/20bit select input Asserting this signal HIGH configures the Receive to accept 10bit FOM inputs. Deasserting this signal LOW configures the Receive to accept 20bit FOM inputs. Encode Word

Signals The Encode Word Module Interface provides all outgoing data and control signals to the FOM. This includes the parallel transmission data and control for both the transmit and receive FOMs. LINK.sub.-- CONTROL Enable FOM Laser Output The Encode assserts this signal HIGH to turn on the FOM transmit laser. LOCK.sub.-- REF Enable FOM Lock to Reference Output The Encode asserts this signal HIGH to force the receive Phase Lock Loop (PLL) to lock to the reference oscillator. LOOP.sub.-- ENABLE FOM Loop-Back Enable Output The Encode asserts this signal HIGH to route FOM output to FOM input. ENCODE.sub.-- TYPE FOM Type Select Input Asserting this signal HIGH configures the Encode to send 10-bit FOM outputs. Deasserting this signal LOW configures the Encode to send 20-bit FOM outputs. LOOP.sub.-- CTL FOM Loop-Back control Input Asserting this signal HIGH enables Encode to generate LOOP.sub.-- ENABLE signal. This a active in both 10 bit and 20 bit mode. LOCK.sub.-- REF.sub.-- CTL FOM Lock to Reference Control Input Asserting this signal HIGH enable Encode to drive LOCK.sub.-- REF output pin in either 10bit or 20bit mode. LASER.sub.-- ON FOM Laser On Input Asserting this signal HIGH enables Encode to drive LINK.sub.-- CONTROL output signal in either 10bit or 20bit mode. FDB.sub.-- OUT[19:0] Fibre Data Out Output These signals contain two 10-bit FCS characters that go to the FOM. See subsection entitled "Encode.sub.-- word Interface Formats" page 4-25 for the pin format. PERROR Parity Error Output The Encode asserts this signal HIGH every time the Encode detects a parity error on either one of the two 8-bit characters just prior to being encoded to 10-bit. Arbitrated Loop The Arbitrated Loop (AL) interface signals are grouped into following Signals subsections: .box-solid. Status .box-solid. Arbitration Control .box-solid. Register Interface Status This group contains signals for indicating the operating status of the Arbitrated Loop. STATUS.sub.-- REG9 Close state Output Asserting this signal HIGH indicates that this port has recognized a Close sequence on the loop. STATUS.sub.-- REG8 LIP Progress State Output Asserting this signal HIGH indicates that the loop initialization is in progress. STATUS.sub.-- REG7 LIP Complete State Output Asserting this signal HIGH indicates that the Loop initialization is complete. STATUS.sub.-- REG6 Opened State Output Asserting this signal HIGH indicates that the both initiator and target Loop ports are ready for communication. STATUS.sub.-- REG5 Open State Output Asserting this signal HIGH indicates that this port is in Open state. This is the state during which a port sends a primitive sequence to open another port for communication. STATUS.sub.-- REG4 Monitoring State Output Asserting this signal HIGH indicates that this port is in Monitoring state. In this state, Link is monitored continuously for various Arbitrated Loop primitive sequences. STATUS.sub.-- REG3 Repetition State Output Assertion of this signal HIGH indicates that this port is in Repetition state. STATUS.sub.-- REG2 Arbitration in Progress for this Output Assertion of this signal indicates that this port is arbitrating for Loop. STATUS.sub.-- REG1 Arbitration in Progress for other Output Assertion of this signal indicates that some other port is arbitrating. STATUS.sub.-- REG0 No Operation Output Loop is idle, LIP is complete and nothing is happening on loop. Arbitration Control This group contains signals for Loop Arbitration. RQST.sub.-- ACC Access Request Input This signal is asserted HIGH by the downstream logic when a communication is needed between this port and another port. ACC.sub.-- GRANTED Access Granted Output This signal is asserted HIGH when arbitration is won by this port. Register Interface This group contains signals to interface AL registers to the downstream logic. AL.sub.-- REG[31:0] Arbitrated Loop register Data Port Input/Output This is a bidirectional 32 bit port used to access registers in AL. Data on this port is synchronous with T.sub.-- WORD.sub.-- CLOCK. AL.sub.-- RD.sub.-- WRITE Arbitrated Loop Read Write Port Inputol Asserting this signal HIGH indicates a READ operation and deasserting this indicates WRITE operation for the selected Register. AL.sub.-- REG.sub.-- SELECT[1:0] Arbitrated Loop register Select Input These bits select one of the registers for read/write operation. Bit[1:0] Register 00 AL.sub.-- TMR.sub.-- REGISTER 01 PORT.sub.-- ACC.sub.-- REGISTER 10-11 Reserved AL.sub.-- TMR.sub.-- REGISTER This is a 32 bit register used for loop time-out functions. PORT.sub.-- ACC.sub.-- REGISTER This is a 32 bit register and contains fiels for AL-PA(8bits) and status conditions(8bits) of other port. Receive Signals The Receive signals are grouped based on functional interface into the following subsections: .box-solid. Receive Word Output .box-